МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ ”ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Розрахункова робота

з дисципліни

“Дискретна математика”

Виконала:

студентка групи КН-114

Гудима Анастасія

Викладач:

Мельникова Н.І.

Львів - 2019

Завдання №1 варіанту №11

Виконати наступні операції над графами:

1) знайти доповнення до першого графу

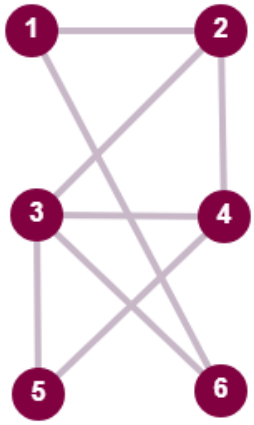
2) об’єднання графів

3) кільцеву сумму G1 та G2

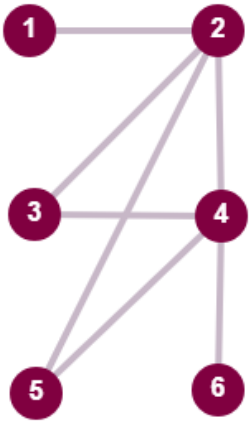
4) розмножити вершину у другому графі

5) виділити підграф А - що скадається з 3-х вершин в G1 і знайти стягнення А в G1 (G1\ A)

6) добуток графів.



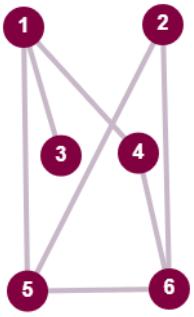
G1



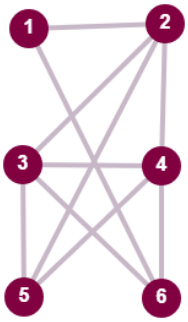
G2

Розв’язок

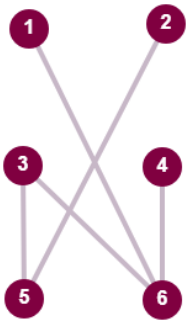
1)



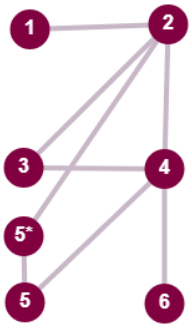
2)



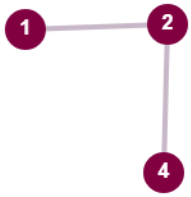
3)



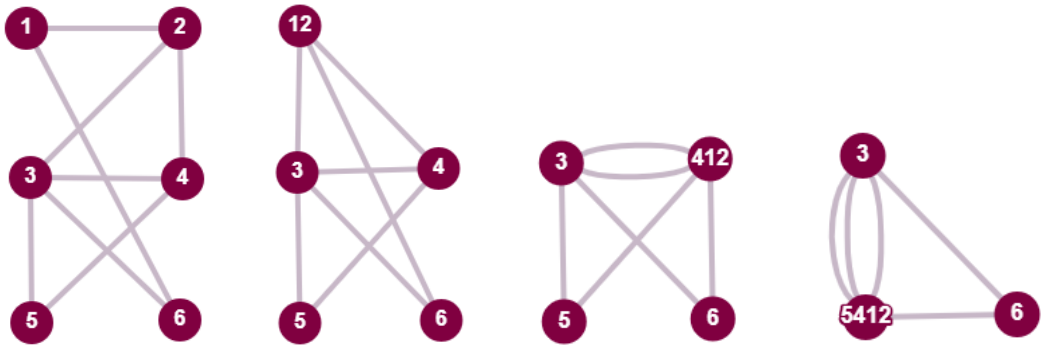
4)



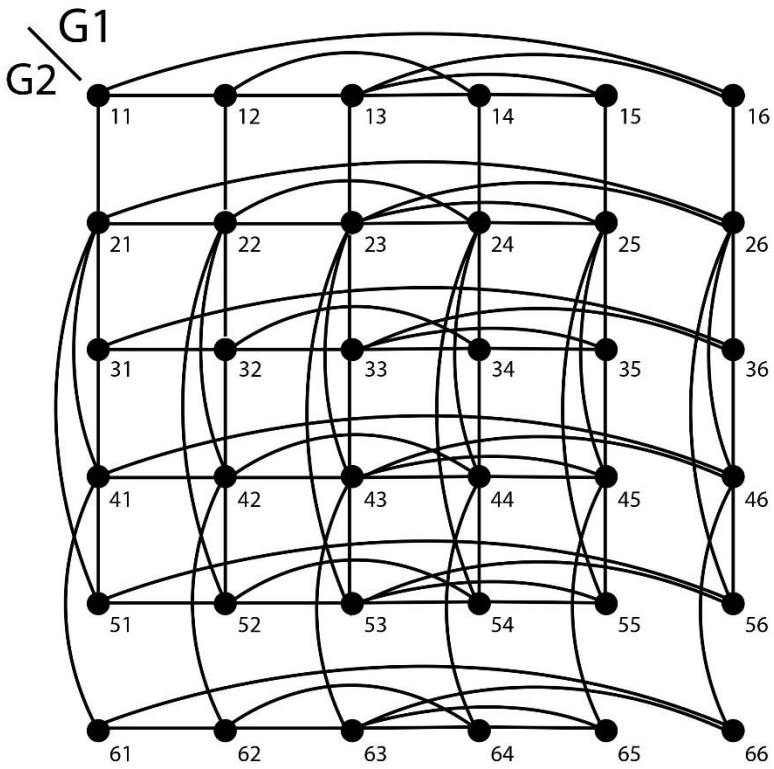
5) підграф А:



Стягнення:

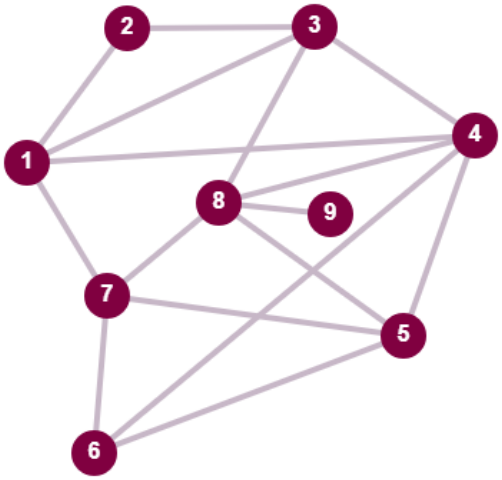


6)



Завдання №2 варіанту №11

Скласти таблицю суміжності для графа.



Розв’язок

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	0	1	1	1	0	0	1	0	0
2	1	0	1	0	0	0	0	0	0
3	1	1	0	1	0	0	0	1	0
4	1	0	1	0	1	1	0	1	0
5	0	0	0	1	0	1	1	1	0
6	0	0	0	1	1	0	1	0	0
7	1	0	0	0	1	1	0	1	0
8	0	0	1	1	1	0	1	0	1
9	0	0	0	0	0	0	0	1	0

Завдання №3 варіанту №11

Для графа з другого завдання знайти діаметр.

Розв’язок

Діаметр – 3.

Завдання №4 варіанту №11

Для графа з другого завдання виконати обхід дерева вглиб.

Розв’язок

Вершина	№	Стек
9	1	9
8	2	98
4	3	984
5	4	9845
6	5	98456
7	6	984567
1	7	9845671
2	8	98456712
3	9	984567123

-	-	98456712
-	-	9845671
-	-	984567
-	-	98456
-	-	9845
-	-	984
-	-	98
-	-	9
-	-	-

```
#include <stdlib.h>
#include <iostream>

using namespace std;

void vhlyb(int start, int n, int **matrix, bool *visited)
{
    cout<<start+1<<" ";
    visited[start]=1;
    for (int i=0; i<=n; i++)
    {
        if ((matrix[start][i]!=0) && (!visited[i]))
        {
            vhlyb(i, n, matrix, visited);
        }
    }
}

void perevirka (int temp)
{
    while (temp != 0 && temp != 1)
        {
            cout<<"Data entered incorrectly. Enter 1 or 0.\n";
            cin >> temp;
        }
}

int main()
{
    int **matrix;
```

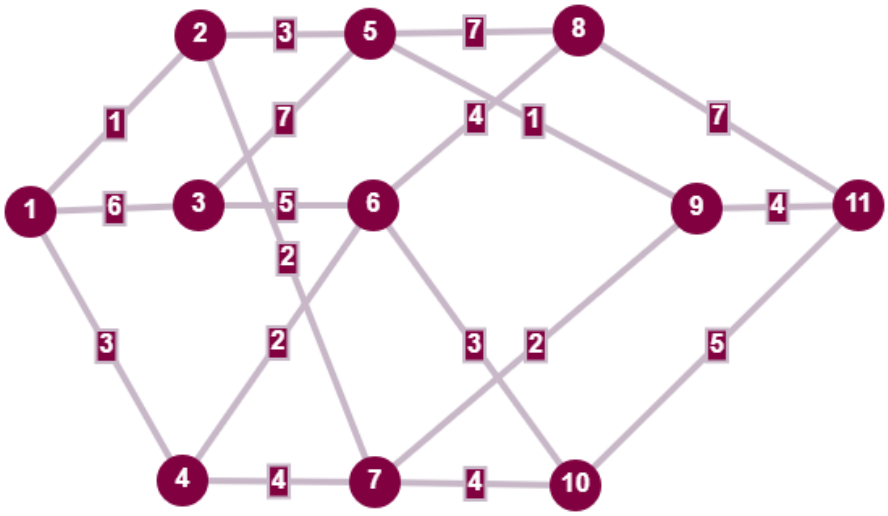
```
    int n, temp;
    int start;

    cout <<"How many vertexes do you want to add? ";
    cin >> n;
    while (n<=0)
    {
        cout<<"Data entered incorrectly. Try one more time.\n";
        cout <<"How many vertexes do you want to add? ";
        cin >> n;
    }
    matrix = (int **) calloc(n, sizeof(int*));

    for (int i=0; i<n; i++)
    {
        matrix[i] = (int *) calloc(n, sizeof(int));
    }
    bool *visited = (bool *) calloc(n, sizeof(bool));

    cout << "Enter your matrix:" << endl;
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
        {
            cin >> temp;
            perevirka (temp);
            if (temp == 1)
            {
                matrix[i][j]=temp;
            }
            else matrix[i][j]=0;
        }
        visited[i]=0;
    }

    cout << "Enter the start vertex: ";
    cin >> start;

    cout << "Your vertexes:\n";
    vhlyb(start-1, n, matrix, visited);
    cout << endl;

    return 0;
}
```

```
How many vertexes do you want to add? 9
Enter your matrix:
0 1 1 1 0 0 1 0 0
1 0 1 0 0 0 0 0 0
1 1 0 1 0 0 0 1 0
1 0 1 0 1 1 0 1 0
0 0 0 1 0 1 1 1 0
0 0 0 1 1 0 1 0 0
1 0 0 0 1 1 0 1 0
0 0 1 1 1 0 1 0 1
0 0 0 0 0 0 0 1 0
Enter the start vertex: 1
Your vertexes:
1 2 3 4 5 6 7 8 9

Process returned 0 (0x0)   execution time : 105.019 s
Press any key to continue.
```

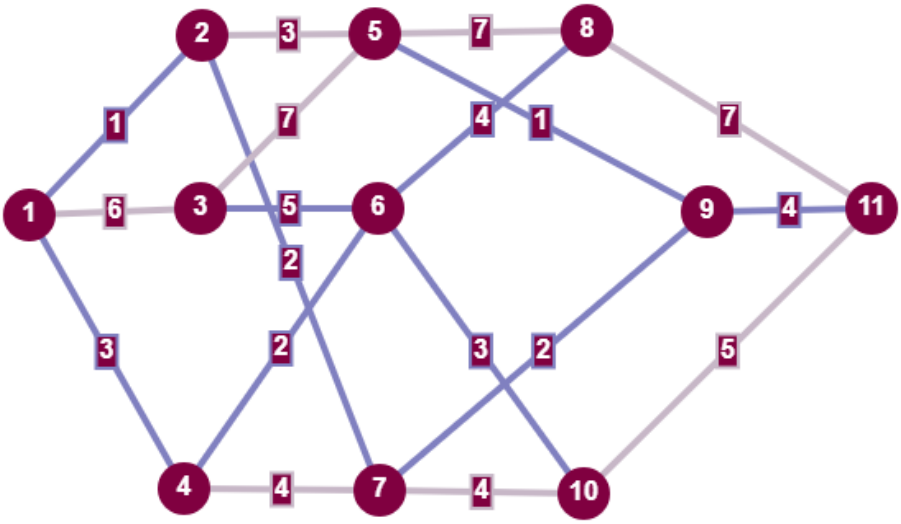
```
How many vertexes do you want to add? -5
Data entered incorrectly. Try one more time.
How many vertexes do you want to add? 3
Enter your matrix:
8
Data entered incorrectly. Enter 1 or 0.
-3
Data entered incorrectly. Enter 1 or 0.
0 1 1
1 0 1
1 1 0
Enter the start vertex: 2
Your vertexes:
2 1 3

Process returned 0 (0x0)   execution time : 60.560 s
Press any key to continue.
```

Завдання №5 варіанту №11

Знайти двома методами (Краскала і Прима) мінімальне остове дерево графа.



Розв’язок



Вага мінімального остового дерева = 27.

Алгоритм Краскала

V={1, 2, 5, 9, 7, 4, 6, 10, 8, 11, 3}

E={(1, 2), (5, 9), (2,7), (4,6), (7,9), (6,10), (1,4), (6,8), (9,11), (3,6)}

```
#include <iostream>
using namespace std;

struct edge
{
    int t1;
    int t2;
    int w;
};

struct edgelist
{
    edge data[30];
    int n;
};

edgelist llist;
edgelist tree;
int G[11][11],n,m;

void sort ()
{
    int i,j;
    edge temp;

    for(i=0;i<llist.n;i++)
    {
        for(j=0;j<llist.n-1;j++)
        {
            if(llist.data[j].w>llist.data[j+1].w)
            {
                temp = llist.data[j];
                llist.data[j] = llist.data[j+1];
                llist.data[j+1] = temp;
            }
        }
    }
}

int find (int belongs[],int vertexno)
{
    return(belongs[vertexno]);
}

void union1 (int belongs[],int c1,int c2)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(belongs[i]==c2)
        {
            belongs[i]=c1;
        }
    }
}

void kraskal ()
{
    int belongs[11],i,j,no1,no2;
    llist.n=0;
```

```
  for(i=0;i<n;i++)
      for(j=0;j<i;j++)
      {
          if(G[i][j]!=0)
          {
              llist.data[llist.n].t1=i;
              llist.data[llist.n].t2=j;
              llist.data[llist.n].w=G[i][j];
              llist.n++;
          }
      }

  sort ();

  for(i=0;i<n;i++)
      belongs[i]=i;

  tree.n=0;

  for(i=0;i<llist.n;i++)
  {
      no1=find(belongs,llist.data[i].t1);
      no2=find(belongs,llist.data[i].t2);

      if(no1!=no2)
      {
          tree.data[tree.n]=llist.data[i];
            tree.n=tree.n+1;
            union1(belongs,no1,no2);
        }
    }
}

void print ()
{
    int i,vaga=0;
    cout<<"\nV1\tV2\tWeight"<<endl;
    for(i=0;i<tree.n;i++)
    {
        cout<<tree.data[i].t2+1<<"\t"<<tree.data[i].t1+1<<"\t"<<tree.data[i].w<<endl;
        vaga=vaga+tree.data[i].w;
    }

    cout<<"\nCost of the spanning tree = "<<vaga<<endl;
    }


int main ()
{
    int ww,i,j,k,c;

    cout<<"Enter the number of vertexes: ";
    cin>>n;

    cout<<"Enter the number of edges: ";
    cin>>m;
```

```
        cout<<"Enter edges(first vertex | second vertex | weight):\n";

        for (int i=0; i<n; i++)
        {
            for (int j=0; j<n; j++)
            {
                G[i][j]=0;
            }
        }

        for(i=0;i<m;i++)
        {
            cout<<"Edge "<<i+1<<": ";
            cin>>k>>c>>ww;
            G[k-1][c-1]=ww;
            G[c-1][k-1]=ww;
        }
        cout<<endl;

    cout<<"Your matrix:\n";
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            cout<<G[i][j]<<" ";
        }
        cout<<endl;
    }

    kraskal();
    print();
}
```

```
Enter the number of vertexes: 11
Enter the number of edges: 18
Enter edges(first vertex | second vertex | weight):
Edge 1: 1 2 1
Edge 2: 1 3 6
Edge 3: 1 4 3
Edge 4: 2 5 3
Edge 5: 2 7 2
Edge 6: 3 5 7
Edge 7: 3 6 5
Edge 8: 4 6 2
Edge 9: 4 7 4
Edge 10: 5 8 7
Edge 11: 5 9 1
Edge 12: 6 8 4
Edge 13: 6 10 3
Edge 14: 7 9 2
Edge 15: 7 10 4
Edge 16: 8 11 7
Edge 17: 9 11 4
Edge 18: 10 11 5
```

```
Your matrix:
0 1 6 3 0 0 0 0 0 0 0
1 0 0 0 3 0 2 0 0 0 0
6 0 0 0 7 5 0 0 0 0 0
3 0 0 0 0 2 4 0 0 0 0
0 3 7 0 0 0 0 7 1 0 0
0 0 5 2 0 0 0 4 0 3 0
0 2 0 4 0 0 0 0 2 4 0
0 0 0 0 7 4 0 0 0 0 7
0 0 0 0 1 0 2 0 0 0 4
0 0 0 0 0 3 4 0 0 0 5
0 0 0 0 0 0 0 7 4 5 0

V1      V2      Weight
1       2       1
5       9       1
4       6       2
2       7       2
7       9       2
1       4       3
6       10      3
6       8       4
9       11      4
3       6       5

Cost of the spanning tree= 27

Process returned 0 (0x0)   execution time : 4.156 s
Press any key to continue.
```

Алгоритм Прима

V={6, 4, 1, 2, 7, 9, 5, 10, 8, 11, 3}

E={(6, 4), (4, 1), (1, 2), (2, 7), (7, 9), (9, 5), (6, 10), (6, 8), (9, 11), (3, 6)}

```cpp
#include <iostream>

using namespace std;

struct edge
{
    int t1;
    int t2;
    int weight;
};

void vvid (edge*p,int n,int m)
{
    cout<<"Enter edges(first vertex | second vertex | weight):\n";
    for (int i=0;i<m;i++)
    {
        cout<<"Edge"<<i+1<<": ";
        cin>>p[i].t1>>p[i].t2>>p[i].weight;
        while (p[i].t1<0 || p[i].t1>n || p[i].t2<0 || p[i].t2>n || p[i].weight<0)
        {
            cout<<"The data entered incorrectly."<<endl;
            cout<<"Try again please."<<endl;
            cin>>p[i].t1>>p[i].t2>>p[i].weight;
        }
    }
}

void bulb(edge* p, int n)
{
```

```
  edge temp;
  for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n - i - 1; j++)
        {
            if (p[j].weight > p[j+1].weight)
            {
                temp = p[j];
                p[j] = p[j+1];
                p[j+1] = temp;
            }
        }
    }
}

bool vkluchene(int* a, int n, int f)
{
  for (int i = 0; i < n; i++)
  {
    if (f == a[i])
    {
      return true;
    }
  }
  return false;
}

bool minn(int w, edge* ed,int m, int* v, int n)
{
  for (int j = 1; j < m; j++)
    {
    if ((((!vkluchene(v, n, ed[j].t1) && vkluchene(v, n, ed[j].t2)) ||
         (vkluchene(v, n, ed[j].t1) && !vkluchene(v, n, ed[j].t2)))
        && ed[j].weight < w)
        {
            return false;
        }
    }
  return true;
}

void pryma (edge*ed,int*v,edge*tree,int n,int m,int&i,int&j)
{
    if (i==n)
    {return;}
    else if(j==n)
    {j=1;}
    if(vkluchene(v, n, ed[j].t1) && vkluchene(v, n, ed[j].t2))
    {
      j++;
      pryma(ed, v, tree, n, m, i, j);
    }
    else if (!vkluchene(v, n, ed[j].t1) && vkluchene(v, n, ed[j].t2)
             && minn(ed[j].weight, ed, m, v, n))
    {
      tree[i-1]=ed[j];
      v[i] = ed[j].t1;
```

```
      j++;
      i++;
      pryma(ed, v, tree, n, m, i, j);

    }
    else if (vkluchene(v, n, ed[j].t1) && !vkluchene(v, n, ed[j].t2)
             && minn(ed[j].weight, ed, m, v, n))
    {
      tree[i-1]=ed[j];
      v[i] = ed[j].t2;

      j++;
      i++;
      pryma(ed, v, tree, n, m, i, j);
    }
    else
    {
      j++;
      pryma(ed, v, tree, n, m, i, j);
    }

}

int main()
{
    int n,m;
    cout<<"How many vertexes do you want to add? ";
    cin>>n;
    cout<<"How many edges do you want to add? ";
    cin>>m;
    cout<<endl;
    edge *ed = new edge[m];
    int *v = new int[n];
    edge *tree = new edge [n-1];
    vvid(ed,n,m);
    bulb(ed,m);
    v[0]=ed[0].t1;
    v[1]=ed[0].t2;
    tree[0]=ed[0];
    int i=2;
    int j=1;
    pryma (ed,v,tree,n,m,i,j);
    cout<<"\nV = {";
    for(int x=0;x<n;x++)
    {
        cout<<v[x]<<",";
    }
    cout<<"}\nE = { ";
    for(int x=0;x<n-1;x++)
    {
        cout<<"("<<tree[x].t1<<","<<tree[x].t2<<") ";
    }
    cout<<"}\n";
    return 0;
}
```

```
How many vertexes do you want to add? 11
How many edges do you want to add? 18

Enter edges(first vertex | second vertex | weight):
Edge1: 1 2 1
Edge2: 1 3 6
Edge3: 1 4 3
Edge4: 2 5 3
Edge5: 2 7 2
Edge6: 3 5 7
Edge7: 3 6 5
Edge8: 4 6 2
Edge9: 4 7 4
Edge10: 5 8 7
Edge11: 5 9 1
Edge12: 6 8 4
Edge13: 6 10 3
Edge14: 7 9 2
Edge15: 7 10 4
Edge16: 8 11 7
Edge17: 9 11 4
Edge18: 10 11 5

V = {1,2,7,9,5,4,6,10,3,8,11,}
E = { (1,2) (2,7) (7,9) (5,9) (1,4) (4,6) (6,10) (6,8) (9,11) (3,6) }

Process returned 0 (0x0)   execution time : 72.387 s
Press any key to continue.
```

Завдання №6 варіанту №11

Розв’язати задачу комівояжера для повного 8-ми вершинного графа методом «іди у найближчий», матриця вагів якого має вигляд:

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	∞	7	6	5	4	3	2	1
2	7	∞	1	5	6	4	2	3
3	6	1	∞	1	5	6	2	3
4	5	5	1	∞	3	2	2	2
5	4	6	5	3	∞	2	2	2
6	3	4	6	2	2	∞	5	5
7	2	2	2	2	2	5	∞	2
8	1	3	3	2	2	5	2	∞

Розв’язок

	2	3	4	5	6	187
2	∞	1	5	6	4	2
3	1	∞	1	5	6	2
4	5	1	∞	3	2	2
5	6	5	3	∞	2	2
6	4	6	2	2	∞	5
187	2	2	2	2	5	∞

	18723	4	5	6
18723	∞	1	5	6
4	1	∞	3	2
5	5	3	∞	2
6	6	2	2	∞

	5	1872346
5	∞	2
1872346	2	∞

```
#include <stdlib.h>
#include <iostream>

using namespace std;

void enter (int n, int **matrix)
{
    cout << "\nEnter weights of edges" << endl;
    cout << "Your matrix: " << endl;
    for (int i=0; i<n; i++)
        for (int j=0; j<n; j++)
            cin >> matrix[i][j];
    cout << endl;
}

void salesman (int n, int start, int **matrix, int *ver)
{
    int temp;
    int min = INT_MAX;
    for (int i=0; i<n; i++)
    {
        ver[i]=start + 1;
        cout << ver[i] << " ";
        for (int j=0; j<n; j++)
```

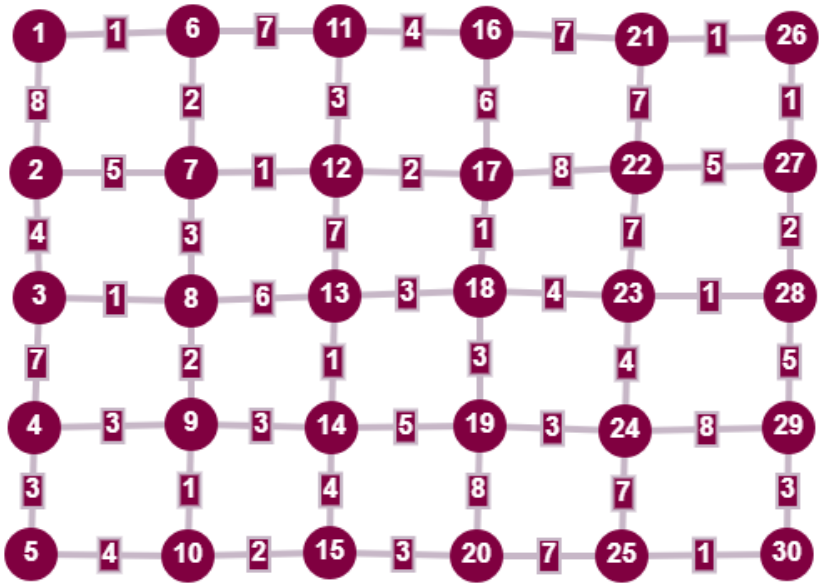
```
        {
            if(matrix[start][j]<min && matrix[start][j]!=0)
            {
                min = matrix[start][j];
                temp=j;
            }
        }
        min = INT_MAX;
        for (int k=0; k<n; k++)
        {
            matrix[start][k]=0;
            matrix[k][start]=0;
        }
        start=temp;
    }
}

int main()
{
    int n;
    int **matrix, *ver;
    int v, start;

    cout << "Enter the number of vertexes: ";
    cin >> n;

    matrix = (int **) calloc(n, sizeof(int*));
    for (int i=0; i<n; i++)
        matrix[i] = (int *)calloc(n, sizeof(int));
    ver = (int *)calloc(n, sizeof(int));

    enter(n, matrix);

    cout << "Enter the start vertex: " << endl;
    cin >> v;

    cout << "The order of vertexes: " << endl;
    start = v-1;

    salesman(n, start, matrix, ver);

    return 0;
}
```

```
Enter the number of vertexes: 8

Enter weights of edges
Your matrix:
0 7 6 5 4 3 2 1
7 0 1 5 6 4 2 3
6 1 0 1 5 6 2 3
5 5 1 0 3 2 2 2
4 6 5 3 0 2 2 2
3 4 6 2 2 0 5 5
2 2 2 2 2 5 0 2
1 3 3 2 2 5 2 0

Enter the start vertex:
1
The order of vertexes:
1 8 4 3 2 7 5 6
Process returned 0 (0x0)   execution time : 47.057 s
Press any key to continue.
```

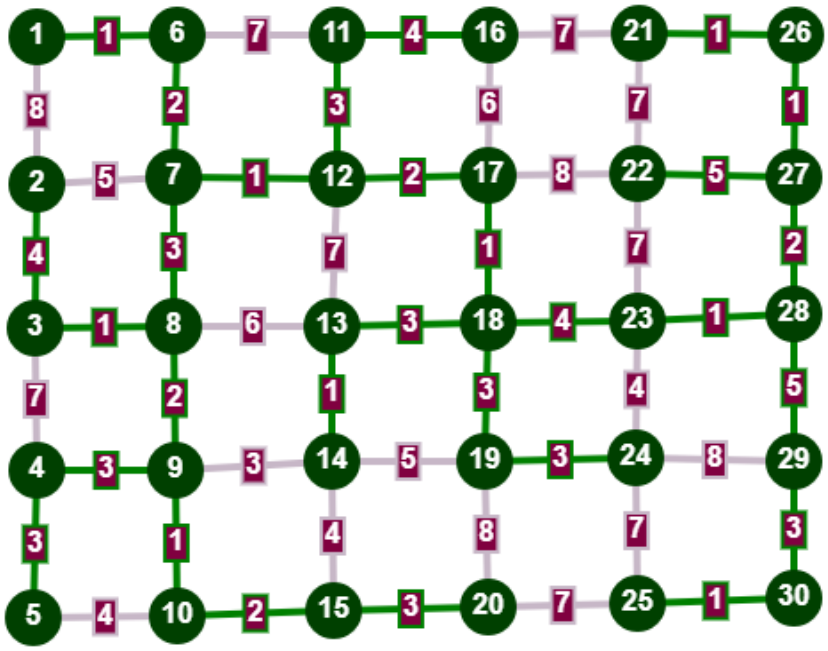
```
Enter the number of vertexes: 8

Enter weights of edges
Your matrix:
0 7 6 5 4 3 2 1
7 0 1 5 6 4 2 3
6 1 0 1 5 6 2 3
5 5 1 0 3 2 2 2
4 6 5 3 0 2 2 2
3 4 6 2 2 0 5 5
2 2 2 2 2 5 0 2
1 3 3 2 2 5 2 0

Enter the start vertex:
5
The order of vertexes:
5 6 4 3 2 7 1 8
Process returned 0 (0x0)   execution time : 16.681 s
Press any key to continue.
```

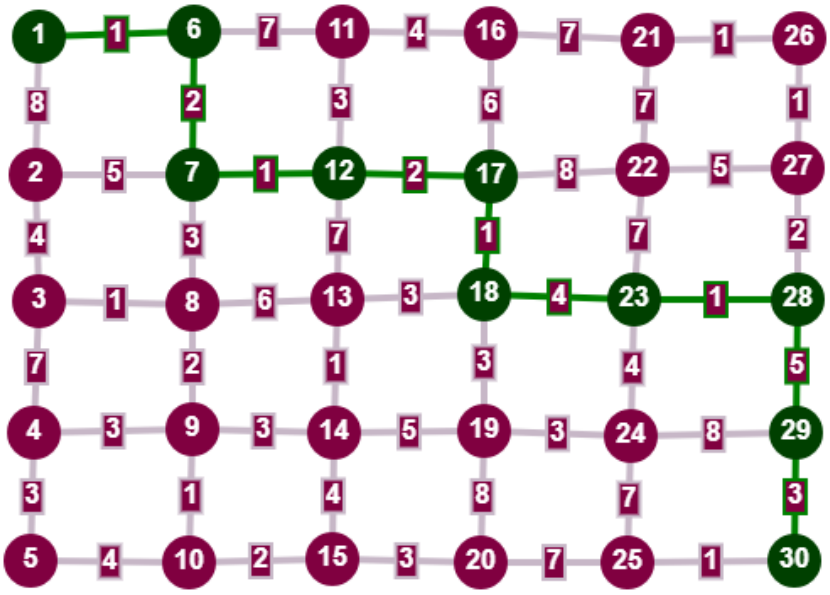
Завдання №7 варіанту №11

За допомогою алгоритму Дейкстри знайти найкоротший шлях у графі між парою вершин V0 і V* .



Розв’язок





Найкоротший шлях – 20.

```cpp
#include <iostream>

using namespace std;

int main()
{
    int n,m;
    cout<<"How many vertexes do you want to add? ";
    cin>>n;
    cout<<"How many edges do you want to add? ";
    cin>>m;
    cout<<endl;

    int s=n;
    int a[s][s];
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
        {
            a[i][j]=0;
        }
    }

    //вводимо потрібні ребра
    int t1,t2,w;
    cout<<"Enter edges(first vertex | second vertex | weight):\n";
    for (int k=0;k<m;k++)
    {
        cout<<"Edge"<<k+1<<": ";
        cin>>t1>>t2>>w;
        while (t1<0 || t1>n || t2<0 || t2>n || w<0)
          {
              cout<<"The data entered incorrectly."<<endl;
              cout<<"Try again please."<<endl;
              cin>>t1>>t2>>w;
          }
          a[t1-1][t2-1]=w;
          a[t2-1][t1-1]=w;
    }
    cout<<endl;

    int sh[n];//мін. шляхи
    int v[n];//пройдені вершини
    int temp, index, minn;
    for (int i = 0; i<n; i++)
    {
        sh[i] = 300000;
        v[i] = 1;
    }
    sh[0] = 0;

    do
    {
        index = 300000;
        minn = 300000;
        for (int i = 0; i<n; i++)
        {
            //проходимо всі вершини
            //якщо вершина не пройдена і вага менша 300000
            if ((v[i] == 1) && (sh[i]<minn))
            {
```

```
                minn = sh[i];
                index = i;
            }
        }

        //додаємо знайдену мінімальну вагу і позначаємо
        if (index != 300000)
        {
            for (int i = 0; i<n; i++)
            {
                if (a[index][i] > 0)
                {
                    temp = minn + a[index][i];
                    if (temp < sh[i])
                    {
                        sh[i] = temp;
                    }
                }
            }
            v[index] = 0;
        }
    } while (index < 300000)

    cout<<"Shortest length from V1 to V"<<n<<" is: "<<sh[n-1]<<endl;
    //пошук шляху
    int vvv[n];
    int kin = n-1;
    vvv[0] = kin + 1;
    int k = 1; //індекс попередньої вершини
    int weight = sh[kin];

    while (kin != 0) //поки не дійдем до поч. вершини
    {
        for (int i = 0; i<n; i++)
        if (a[kin][i] != 0)//перевірка чи є таке ребро
        {
            int temp = weight - a[kin][i]; //шукаємо вагу попередньої вершини
            if (temp == sh[i])
            {
                weight = temp;
                kin = i;
                vvv[k] = i + 1;
                k++;
            }
        }
    }
    //вивід шляху
    cout<<"\nThe shortest way:\n";
    for (int i = k - 1; i >= 0; i--)
    {
        cout<<vvv[i]<<"  ";
    }
    cout<<endl;

    return 0;
}
```

```
How many vertexes do you want to add? 30
How many edges do you want to add? 49

Enter edges(first vertex | second vertex | weight):
Edge1: 1 6 1
Edge2: 1 2 8
Edge3: 2 7 5
Edge4: 2 3 4
Edge5: 3 8 1
Edge6: 3 4 7
Edge7: 4 9 3
Edge8: 4 5 3
Edge9: 5 10 4
Edge10: 6 11 7
Edge11: 6 7 2
Edge12: 7 12 1
Edge13: 7 8 3
Edge14: 8 13 6
Edge15: 8 9 2
Edge16: 9 14 3
Edge17: 9 10 1
Edge18: 10 15 2
Edge19: 11 16 4
Edge20: 11 12 3
Edge21: 12 17 2
Edge22: 12 13 7
Edge23: 13 18 3
Edge24: 13 14 1
Edge25: 14 19 5
Edge26: 14 15 4
```

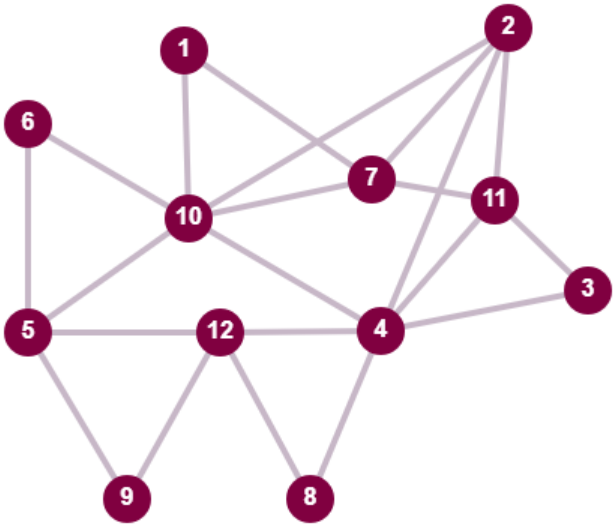
```
Edge27: 15 20 3
Edge28: 16 21 7
Edge29: 16 17 6
Edge30: 17 22 8
Edge31: 17 18 1
Edge32: 18 23 4
Edge33: 18 19 3
Edge34: 19 24 3
Edge35: 19 20 8
Edge36: 20 25 7
Edge37: 21 26 1
Edge38: 21 22 7
Edge39: 22 27 5
Edge40: 22 23 7
Edge41: 23 28 1
Edge42: 23 24 4
Edge43: 24 29 8
Edge44: 24 25 7
Edge45: 25 30 1
Edge46: 26 27 1
Edge47: 27 28 2
Edge48: 28 29 5
Edge49: 29 30 3

Shortest length from V1 to V30 is: 20

The shortest way:
1  6  7  12  17  18  23  28  29  30

Process returned 0 (0x0)   execution time : 366.085 s
Press any key to continue.
```

Завдання №8 варіанту №11

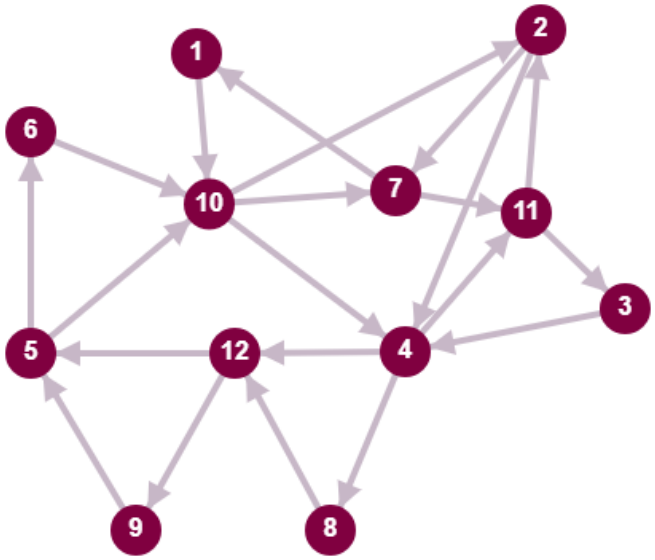
Знайти ейлеровий цикл в ейлеровому графі двома методами:

а) Флері;

б) елементарних циклів.



Розв’язок



а) $10 \to 7 \to 11 \to 3 \to 4 \to 8 \to 12 \to 9 \to 5 \to 6 \to 10 \to 4 \to 12 \to 5 \to 10 \to$
$\to 2 \to 4 \to 11 \to 2 \to 7 \to 1 \to 10$

```cpp
#include <iostream>
#include <cstdio>

using namespace std;

int k;
int cykl[100];

void Search(int n, int v, int matrix [12][12])
{
    int i;
    for(i = 0; i < n; i++)
        if(matrix[v][i])
        {
            matrix[v][i] = matrix[i][v] = 0;
            Search(n,i,matrix);
        }
    cykl[++k] = v+1;
}

int main()
{
    int n = 12;
    int i,s,j,beg_v,temp;
    int matrix [12][12];

    //вводимо матрицю
    cout<<"Enter matrix 12/12:\n";
    for (int i=0; i<n; i++)
    {
          for (int j=0; j<n; j++)
          {
              cin >> temp;
              while (temp != 0 && temp != 1)
              {
                  cout<<"Data entered incorrectly. Enter 1 or 0.\n";
                  cin >> temp;
              }
              if (temp == 1)
              {
                  matrix[i][j]=temp;
              }
              else matrix[i][j]=0;
          }
      }
      int T = 1;

      //перевірка чи граф ейлеровий
      for(i = 0; i < n; i++)
      {
          s = 0;
          for(j=0; j<n; j++)
          {
              s += matrix[i][j];//рахуємо всі 1 в матриці
          }
          if(s%2!=0)//якщо граф не ейлеровий
          {T = 0;}
      }
      k = -1;
      cout << "\nEnter start vertex: ";
```

```
    cin >> beg_v;
    while (beg_v<=0)
    {
        cout<<"Data entered incorrectly. Try one more time.\n";
        cout << "\nEnter start vertex: ";
        cin >> beg_v;
    }
    beg_v-=1;
    if(T==1)//якщо граф існує
    {
        Search (n,beg_v,matrix);
        for(j=0; j<=k; j++)
            cout<<cykl[j]<<" ";
    }
    else
        cout<<"It is not Eulerian graph.\n";
    return 0;
}
```

```
Enter matrix 12/12:
0 0 0 0 0 0 1 0 0 1 0 0
0 0 0 1 0 0 1 0 0 1 1 0
0 0 0 1 0 0 0 0 0 0 1 0
0 1 1 0 0 0 0 1 0 1 1 1
0 0 0 0 0 1 0 0 1 1 0 1
0 0 0 0 1 0 0 0 0 1 0 0
1 1 0 0 0 0 0 0 0 1 1 0
0 0 0 1 0 0 0 0 0 0 0 1
0 0 0 0 1 0 0 0 0 0 0 1
1 1 0 1 1 1 1 0 0 0 0 0
0 1 1 1 0 0 1 0 0 0 0 0
0 0 0 1 1 0 0 1 1 0 0 0

Enter start vertex: 10
10 6 5 12 9 5 10 7 11 4 12 8 4 10 2 11 3 4 2 7 1 10
Process returned 0 (0x0)   execution time : 11.231 s
Press any key to continue.
```

```
Enter matrix 12/12:
0 0 0 0 0 0 1 0 0 1 0 0
0 0 0 1 0 0 1 0 0 1 1 0
0 0 0 1 0 0 0 0 0 0 1 0
0 1 1 0 0 0 0 1 0 1 1 1
0 0 0 0 0 1 0 0 1 1 0 1
0 0 0 0 1 0 0 0 0 1 0 0
1 1 0 0 0 0 0 0 0 1 1 0
0 0 0 1 0 0 0 0 0 0 0 1
0 0 0 0 1 0 0 0 0 0 0 1
1 1 0 1 1 1 1 0 0 0 0 0
0 1 1 1 0 0 1 0 0 0 0 0
0 0 0 1 1 0 0 1 1 0 0 5
Data entered incorrectly. Enter 1 or 0.
-2
Data entered incorrectly. Enter 1 or 0.
0

Enter start vertex: 20
Data entered incorrectly. Try one more time.

Enter start vertex: -3
Data entered incorrectly. Try one more time.

Enter start vertex: 1
1 10 6 5 12 9 5 10 7 11 4 12 8 4 10 2 11 3 4 2 7 1
Process returned 0 (0x0)   execution time : 17.667 s
Press any key to continue.
```

б) Цикли:

1) $10 \to 2 \to 4 \to 12 \to 5 \to 6 \to 10$

2) $5 \to 10 \to 4 \to 8 \to 12 \to 9 \to 5$

3) $1 \to 10 \to 7 \to 1$

4) $2 \to 7 \to 11 \to 2$

5) $4 \to 11 \to 3 \to 4$

Завдання №9 варіанту №11

Спростити формулу (привести їх до скороченої ДНФ).

$\bar{x}\, y \vee \bar{y}\, \bar{z}\, x$

Розв’язок

$\bar{x}\, y \vee \bar{y}\, \bar{z}\, x = \bar{x}\,(y \vee \bar{y}\, \bar{z}\, x) = \bar{x}\,(y\, \bar{z}\, x) = \bar{x}\, y\, \bar{z}$